Tampereen yliopisto
Tampere University

TIE-02306

Introduction to Software Engineering

5 credit units

04-UML-ItSE-2019-v3

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. **Basic UML Diagrams (”Class”, Use Case, Navigation)**
5. UML diagrams, in more detail
6. Different software systems
7. Life Cycle models
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

4. Basic UML Diagrams

- **Entity diagram** = **context diagram**, ”class diagram” (FI: käsitekaavio, tietoyhteyskaavio) **terminogoly is not coherent**
- **Use Case diagram** and User Stories
- **Navigation diagram (not officially UML)**

Those three diagrams you need at project assignment (exercise work).

Tampereen yliopisto
Tampere University

Current at course (w 39)

- we have nine project groups
- **WE4 groups this week are on WED and THU**
- after those we think next week WE5 groups
- **to WE5 BYOC (bring your own computers); Dia tool**
- **Not all Trello boards are ready yet… invite staff**

But at first InnoEvent (04-08.11.2019) info www.innoevent.fi

Tampereen yliopisto
Tampere University

short history of diagramming

Diagrams were introducted early in 1970s to help at requirements, when customers could not understand (technical) requirements specifications or did not care to read thick paper documents.

At that time there were no graphical displays, so pencil and paper were the tools.

In 1990s graphical diagramming software were becoming usual (but expensive !), and with mouse it was somehow easy to draw and **modify** diagrams.

At that time ”every respected guru” designed his own diagramming style, and some of those methods were supported by a tool.

General and cheap drawing/diagramming tool was Windows Paint. But if you moved a box, the attached line did not move with. :-)

Nowadays technology has evolved at huge steps, with any diagramming tool you can draw many different kind of diagrams. There are also some ”rules” at tools, so that you ”must” follow some guidelines (automated checking).

But you still **have to think yourself** what to draw. About some system, you can draw varying diagrams, so you have to decide how you model the system.

So, both text and diagrams are needed at requirements specification.

Tampereen yliopisto
Tampere University

UML = Unified Modeling Language

This specification defines the Unified Modeling Language (UML), revision 2. **The objective of UML is to provide system architects, software engineers, and software developers with tools for analysis, design, and implementation of software-based systems as well as for modeling business and similar processes.**

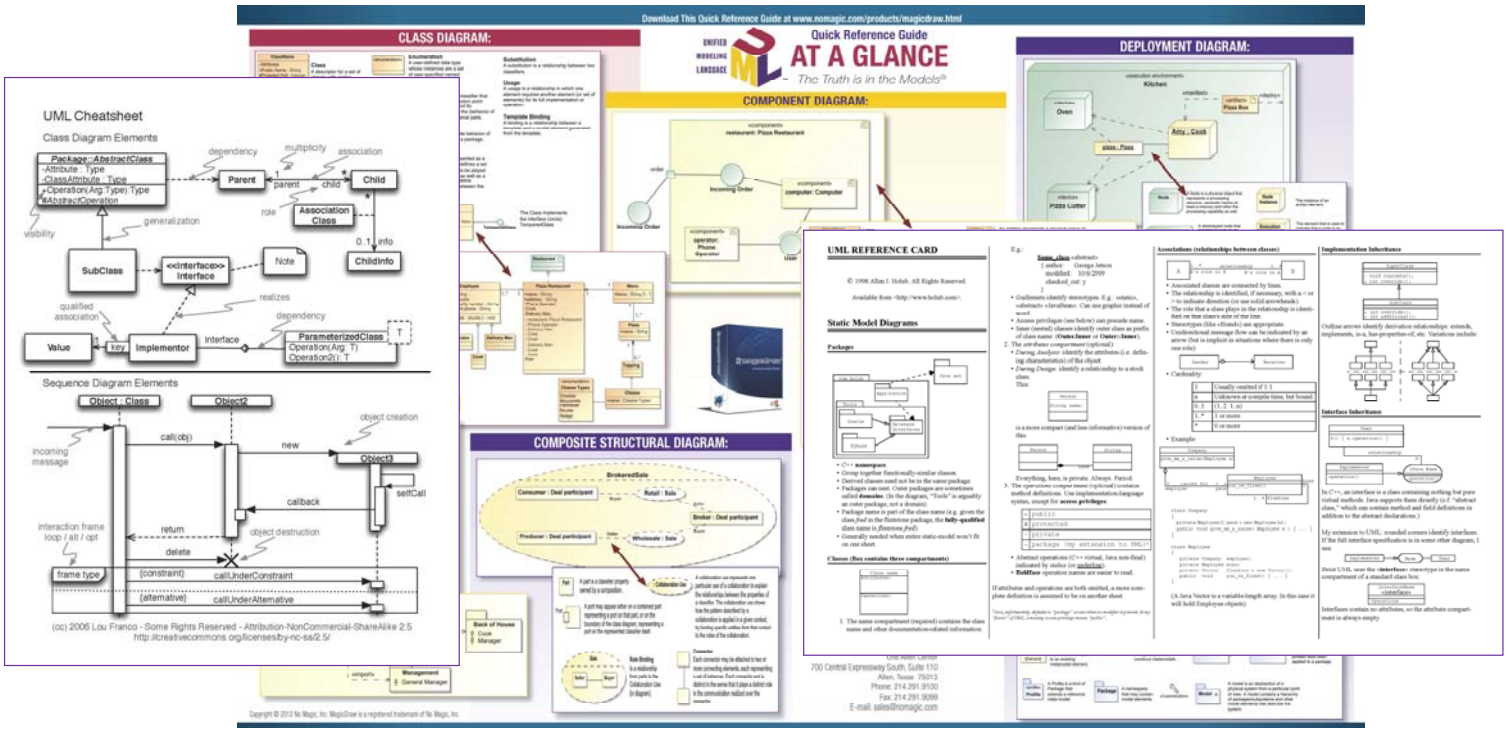
The initial versions of UML (UML 1) originated with three leading object-oriented methods (Booch, OMT, and OOSE), and incorporated a number of best practices from modeling language design, object-oriented programming, and architectural description languages. Relative to UML 1, this revision of UML has been enhanced with significantly more precise definitions of its abstract syntax rules and semantics, a more modular language structure, and a greatly improved capability for modeling large-scale systems.

One of the primary goals of UML is to advance the state of the industry by enabling object visual modeling tool interoperability. However, to enable meaningful exchange of model information between tools, agreement on semantics and syntax is required.

Currently UML is the ”industry standard” diagramming method, by OMG.

However, agree the diagramming style in your project, so that all parties understand it.

There are many UML reference cards and cheat sheets available



Three basic diagrams

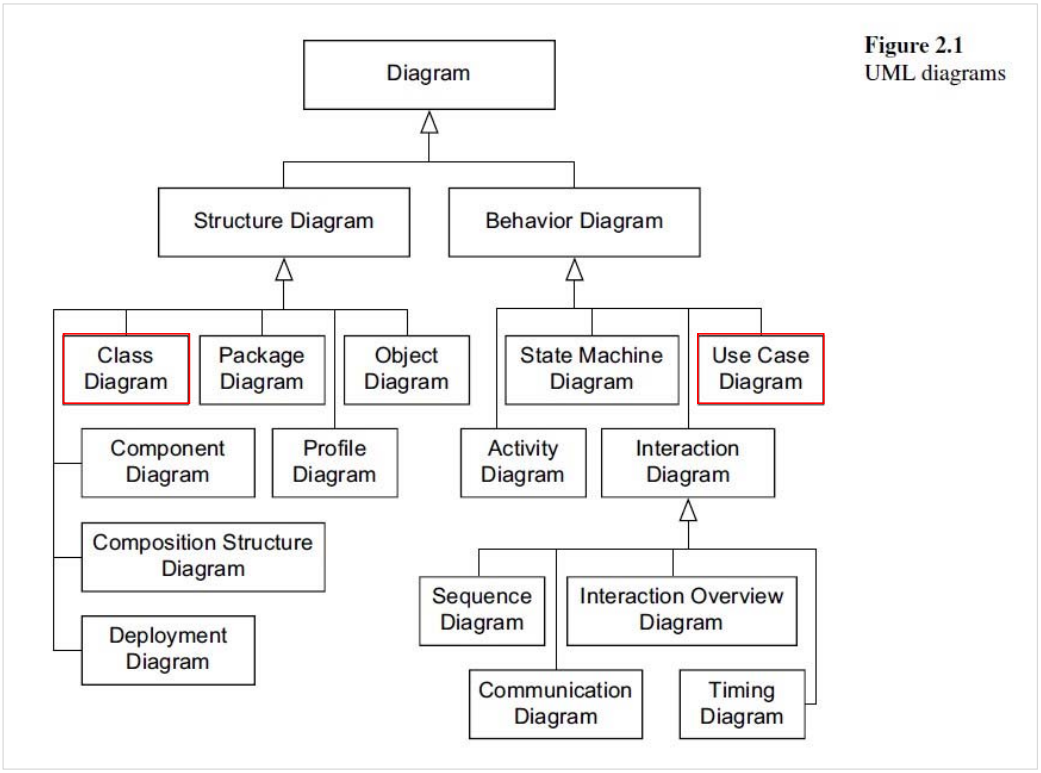
- **Entity diagram** = **context diagram** (conceptual diagram, domain model, class diagram) shows **entities** and their **connections**/relations (associations) in the system. This does not reveal what are the functionalities, and in which order functionalities (should) happen.

- **Use Case diagram** tells **user** groups and **functionalities** (actions). One functionality is explained in detail in textual **User Story**. This does not reveal what data moves in system and in which order functionalities (should/can) happen.

- **Navigation diagram** (display map, menu hierarchy) figures out **how user can navigate** (move) inside the program. This does not reveal who (user group) does (should/can) what, and what data moves in the system.

All these are used in the requirements phase, first in general level and then updated with more detail. Navigation diagram would be useful later in User Guide/Manual.

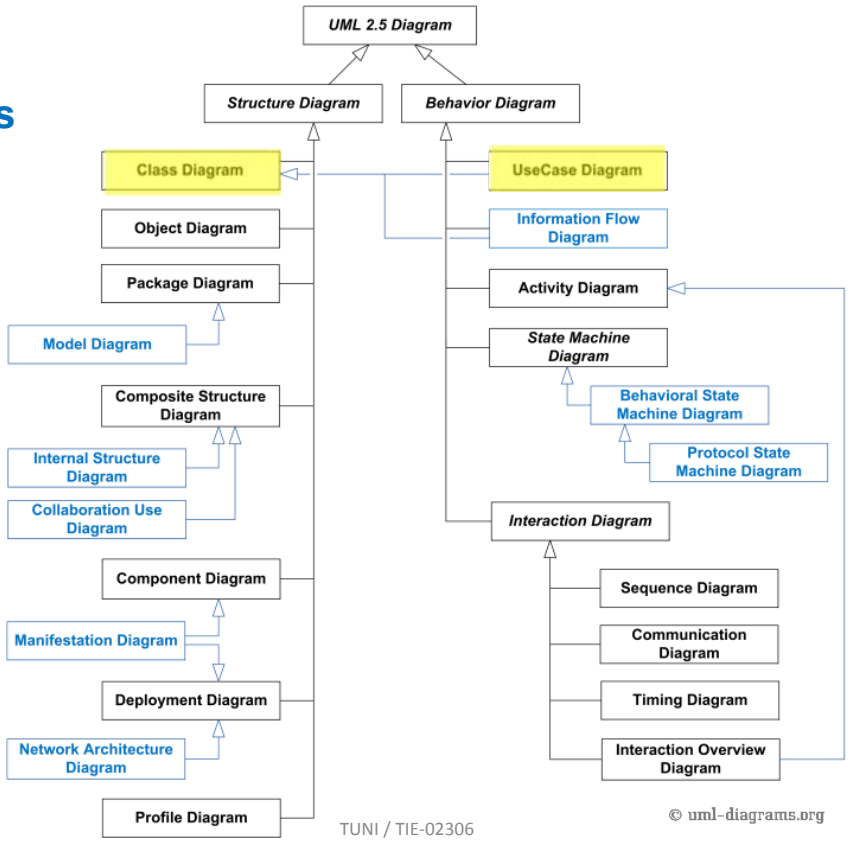
UML diagram types



[UML at classroom, 2015]

UML 2.5 diagrams (2015)

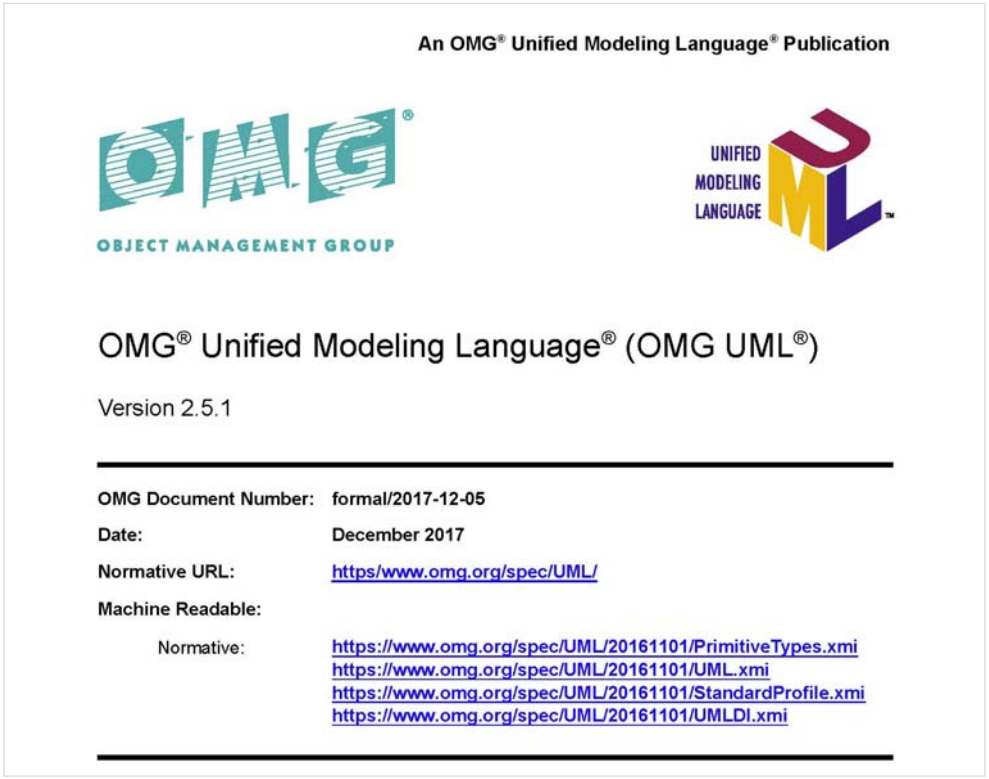
UML diagram types



www.uml-diagrams.org/

UML 2.5.1 (2017) specification

www.omg.org/spec/UML/2.5.1/PDF



An OMG® Unified Modeling Language® Publication

OBJECT MANAGEMENT GROUP

UNIFIED MODELING LANGUAGE

OMG® Unified Modeling Language® (OMG UML®)

Version 2.5.1

OMG Document Number:	formal/2017-12-05
Date:	December 2017
Normative URL:	https/www.omg.org/spec/UML/
Machine Readable:	
Normative:	https://www.omg.org/spec/UML/20161101/PrimitiveTypes.xmi https://www.omg.org/spec/UML/20161101/UML.xmi https://www.omg.org/spec/UML/20161101/StandardProfile.xmi https://www.omg.org/spec/UML/20161101/UMLDI.xmi

Entity / context (conceptual) diagram

In many cases, also in Finnish, this diagram type is called ”Class Diagram”, just because it is drawn by ”class diagram” symbols and functionality in tools.

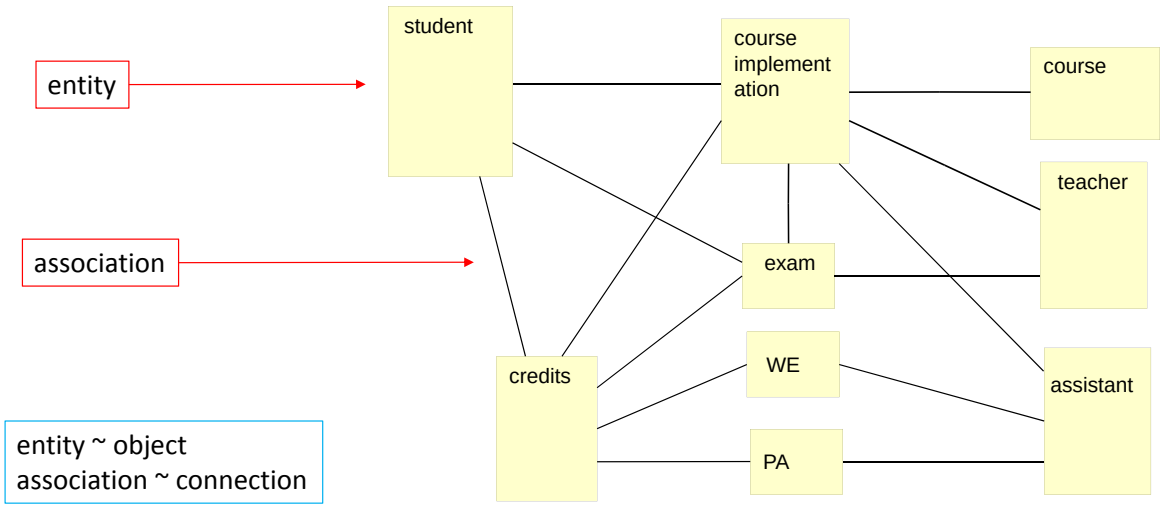
This diagram type is the very first used when starting a project. By entity (or context) diagram you get an overall idea about the application area and environment. It is something like a mind-map.

The idea is to first draw (write) all entities you know, and then connect together those which are related or affect each other. After that, you think carefully which entities are actually needed at the system, and which you may drop out.

By the way, at this point it makes sense to make a (data) **dictionary**, to avoid misunderstandings. For example, at university, what ”class” or ”course” means ?

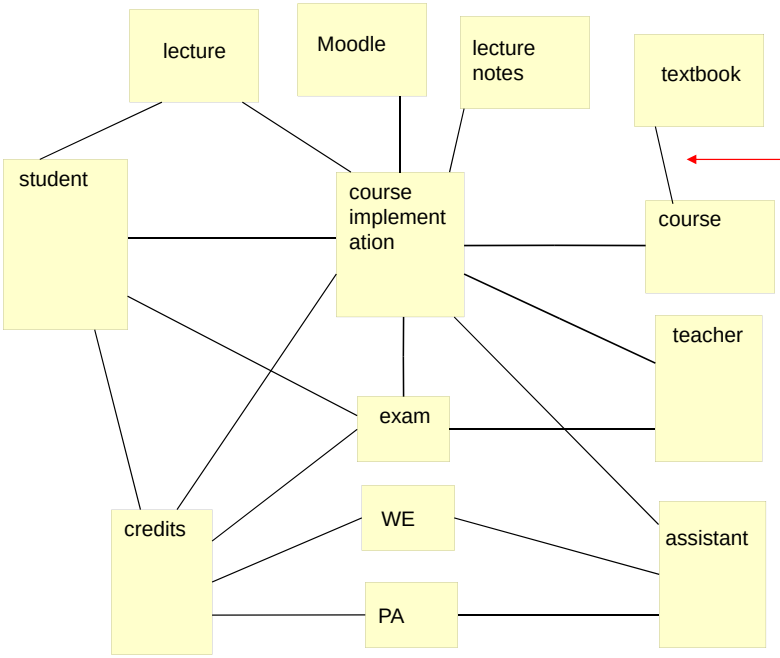
(Actually class diagram is used in design phase, helping to define which attributes and operations an object should have in code.)

Entity / Context diagram, about University course.
What is still missing ?



This diagram does not yet include all necessary entities nor associations (connections).

Entity / Context diagram.



Usually no arrowhead in association lines. But some tools draw only arrows (from --> to).

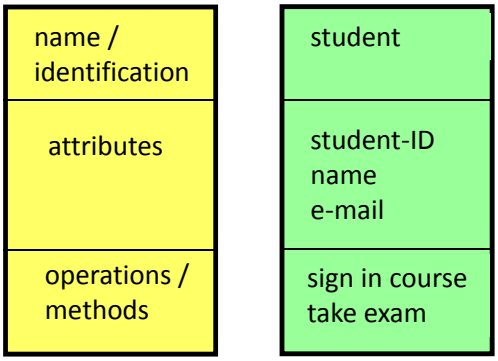
This diagram does not even yet include all necessary entities nor associations (connections) to be complete.

Entity / Context (class) diagram

Terminilogy is not coherent, may vary depending on the speaker.

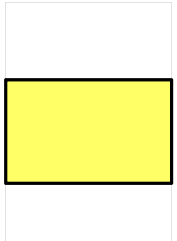
- **entity, instance, object, class,…**

Diagramming tools draw classes as three-part boxes.
Do not care about the third (lower) part.



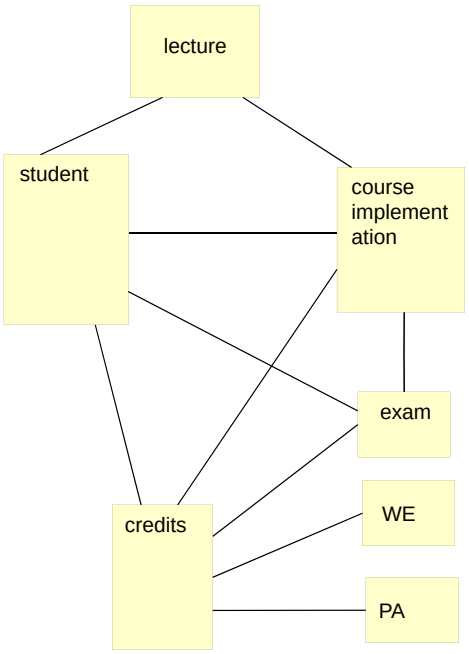
attributes

(middle part)

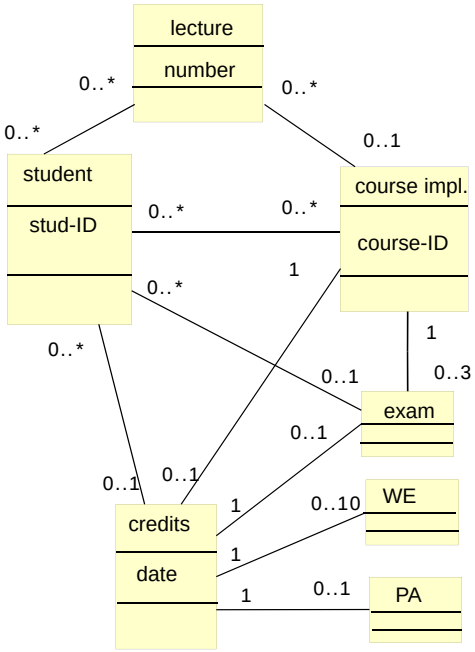


class symbol

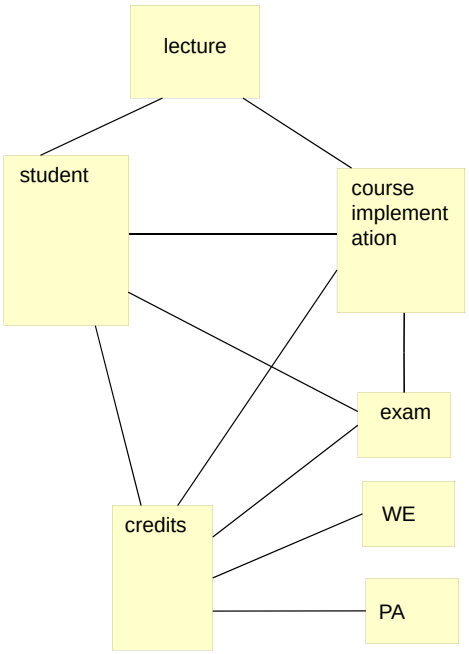
Entity diagram (FI: käsitekaavio)



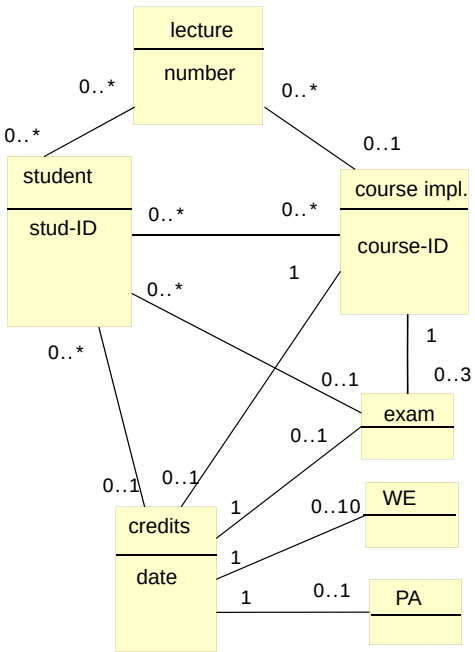
Class diagram (FI: luokkakaavio)



Entity diagram (FI: käsitekaavio)

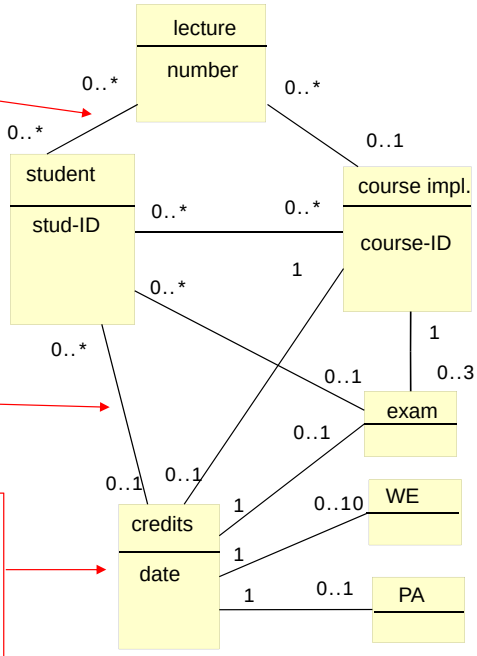


(FI: tietoyhteyskaavio)



Multiplicities (FI: lukumääräsuhteet)

Student takes part or not to one or more lectures = 0..* 0..*
At lectures there are zero or more students.



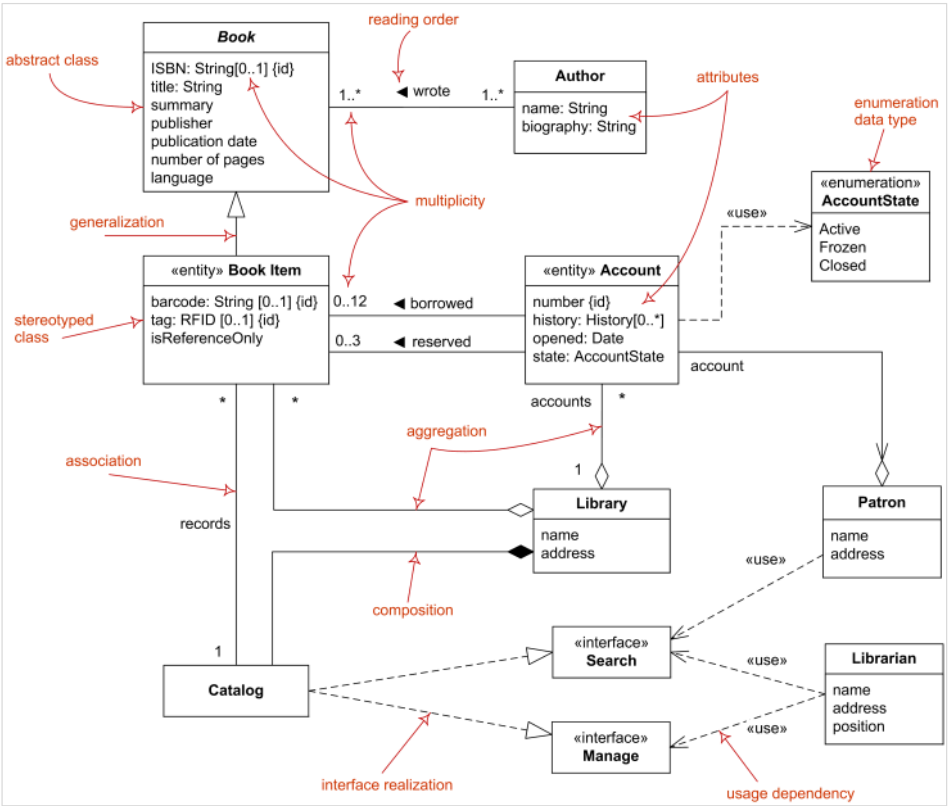
Zero or more students has credits, Student has one or has not (zero) credits.

To one (student's) credits belong
zero or no exam,
zero to ten weekly exercises
zero or one project assignment.
All exam, WE, PA info is connected to some credits.
Credits belong to one course implementation.

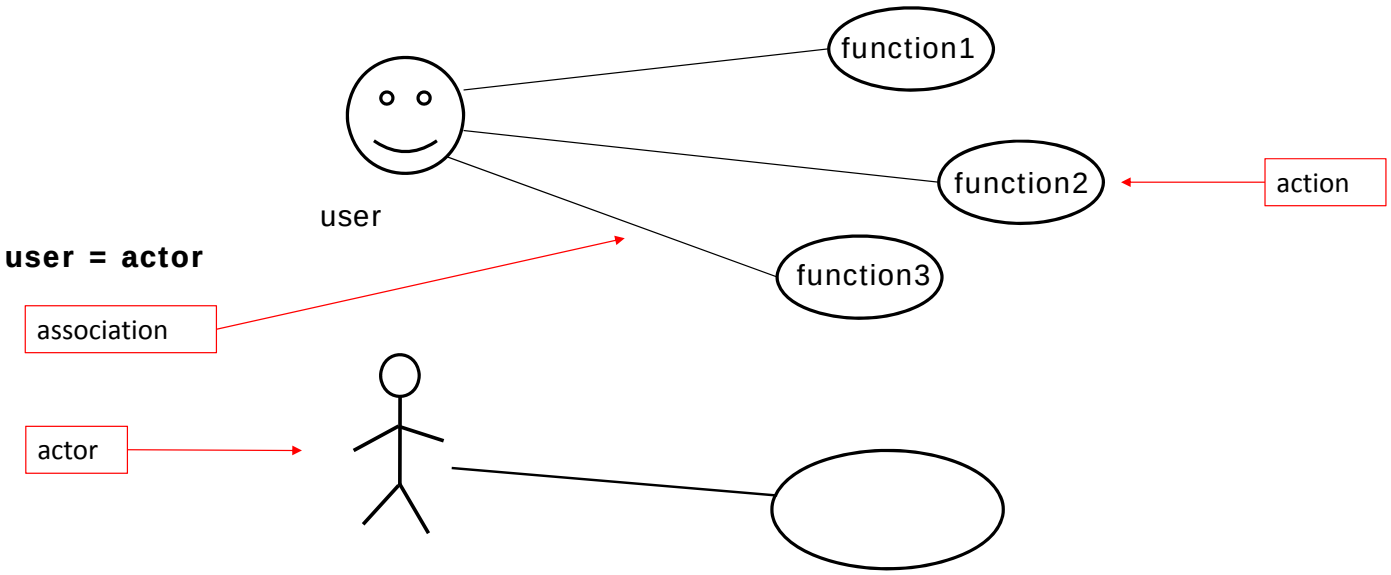
Tampereen yliopisto
Tampere University

Not all the details which are in this example are needed in your PA.

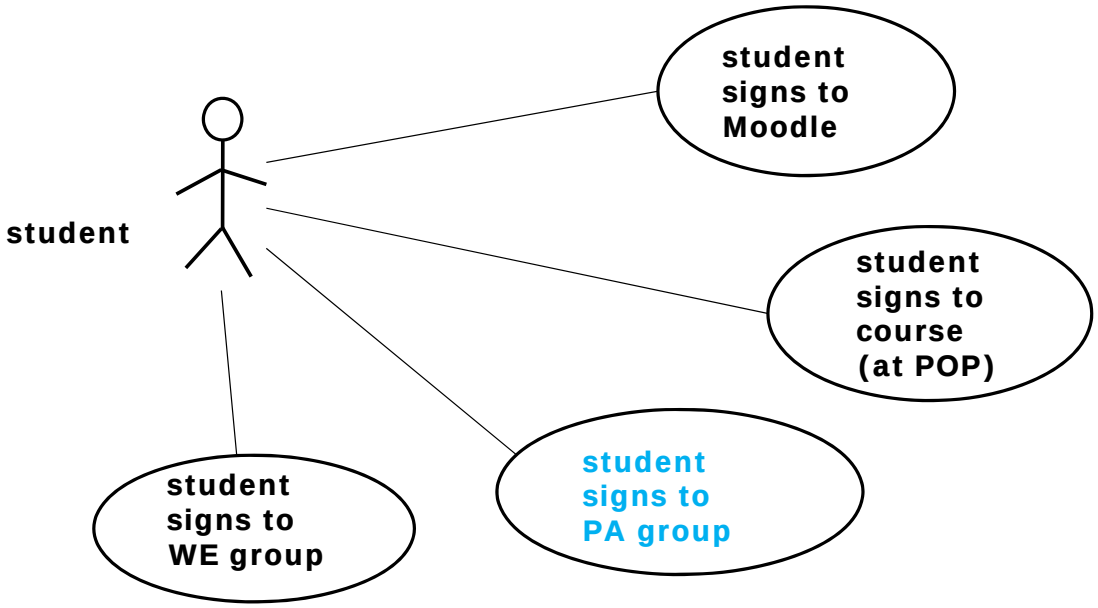
[www.uml-diagrams.org]



Use Case diagram is usually made at the first phases of project, to help at requirements gathering



Use Case diagram, university course example

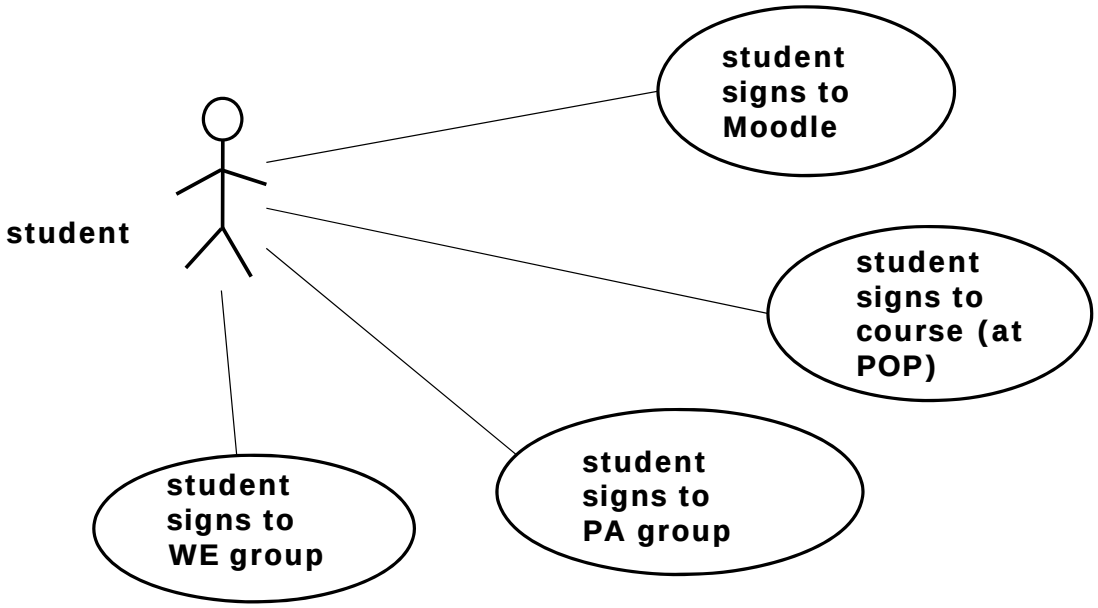


User Story (Use Case), one function at Use Case diagram

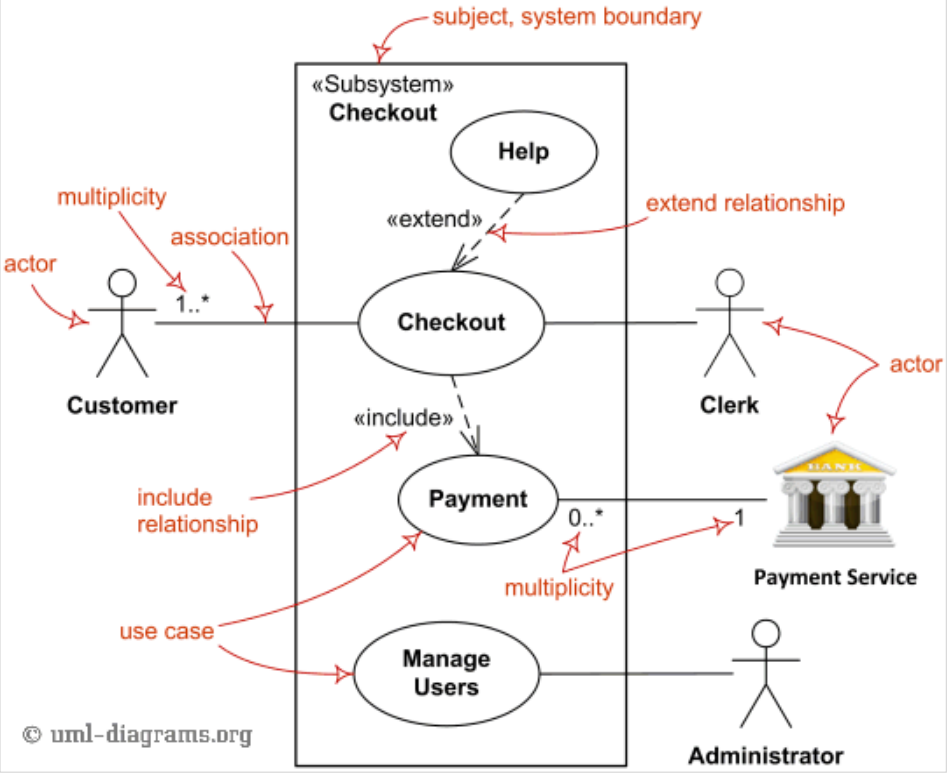
User story for ”Student signs to PA group”, student signs for project assignment (exercise work):

- **Actor:** student.
- **Start state**: Student is signed to course at POP. Student is signed at Moodle. Student has selected course implementation (e.g. ItSE 2019).
- **Actions**: Student selects signing to PA. Student looks at groups and selects one. Students signs to that group. Student gets confirmation message.
- **End state**: student is signed at one PA group.
- **Exceptions**: No ”free seats” at any PA group, all groups are full. Session timeout.
- **Priority**: required.

Use Case diagram, you can not see from that
- dependencies between entities
- what data flows or is in the system
- chronology (order) for the functions.

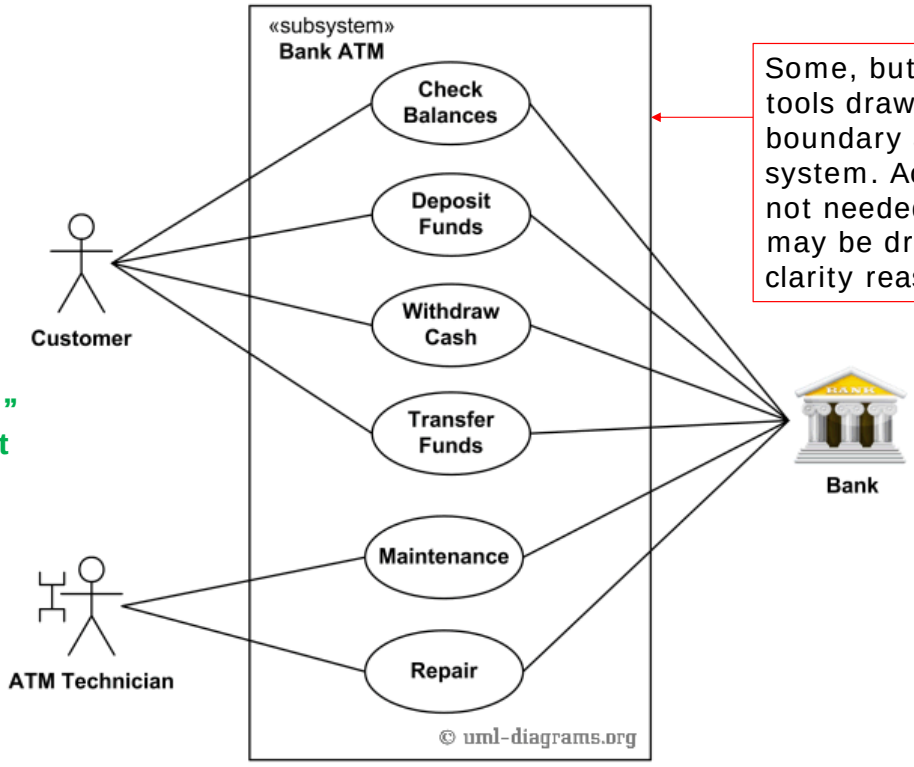


Do not use extend or include, if you are not absolutely sure what those mean.



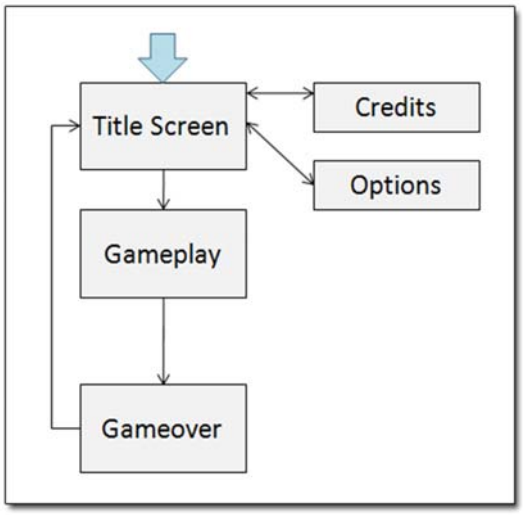
An example of use case diagram for Bank ATM subsystem - top level use cases.

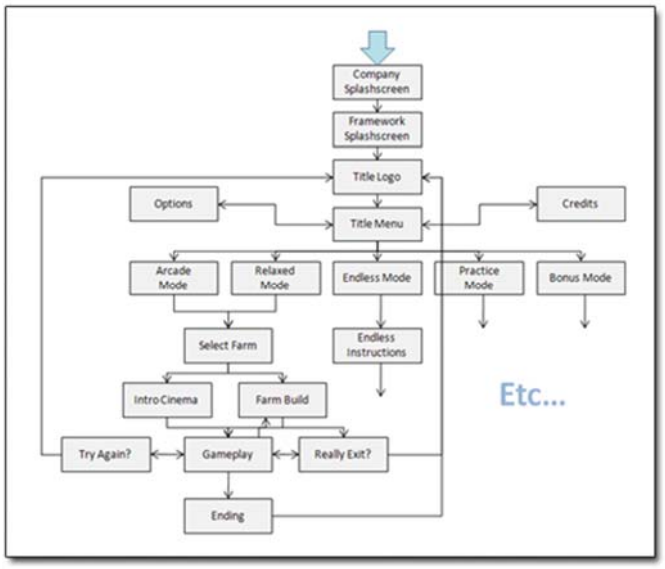
This may look ”childish” simple, but it tells what are the users, and which user groups are doing what.



Some, but not all, tools draw a boundary around the system. Actually it is not needed, but it may be drawn for clarity reasons.

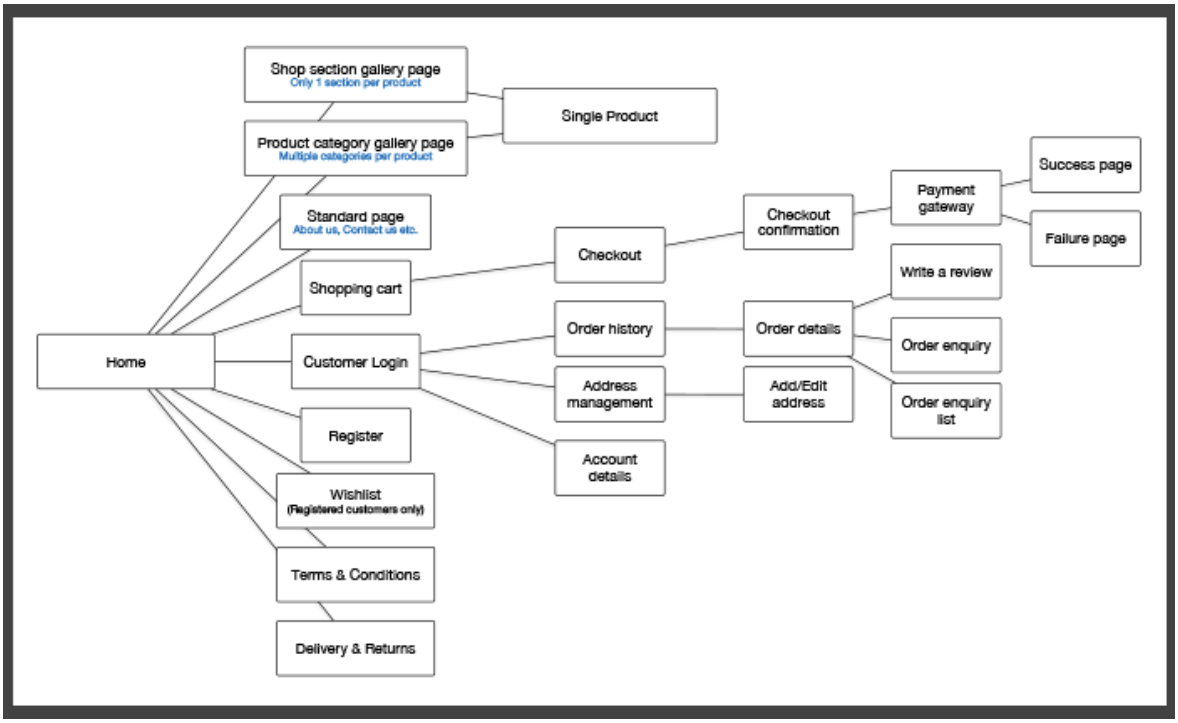
Navigation diagram / map / grid / chart (feature tree)





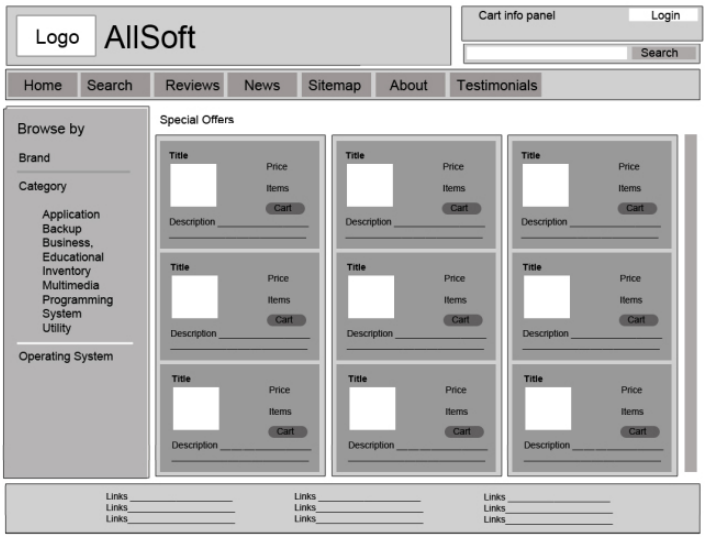
[rivermanmedia.com/gui-design-tip-the-navigation-grid/]

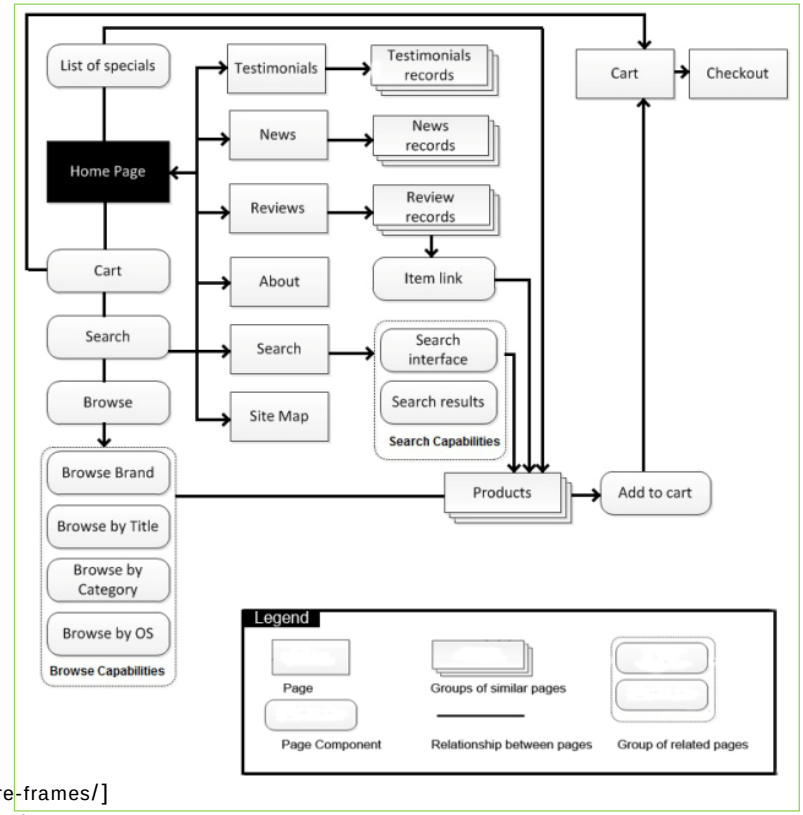
Web shop navigation



[www.nexusdp.co.uk/blog/2016/04/8-step-guide-to-creating-a-successful-small-business-website/]

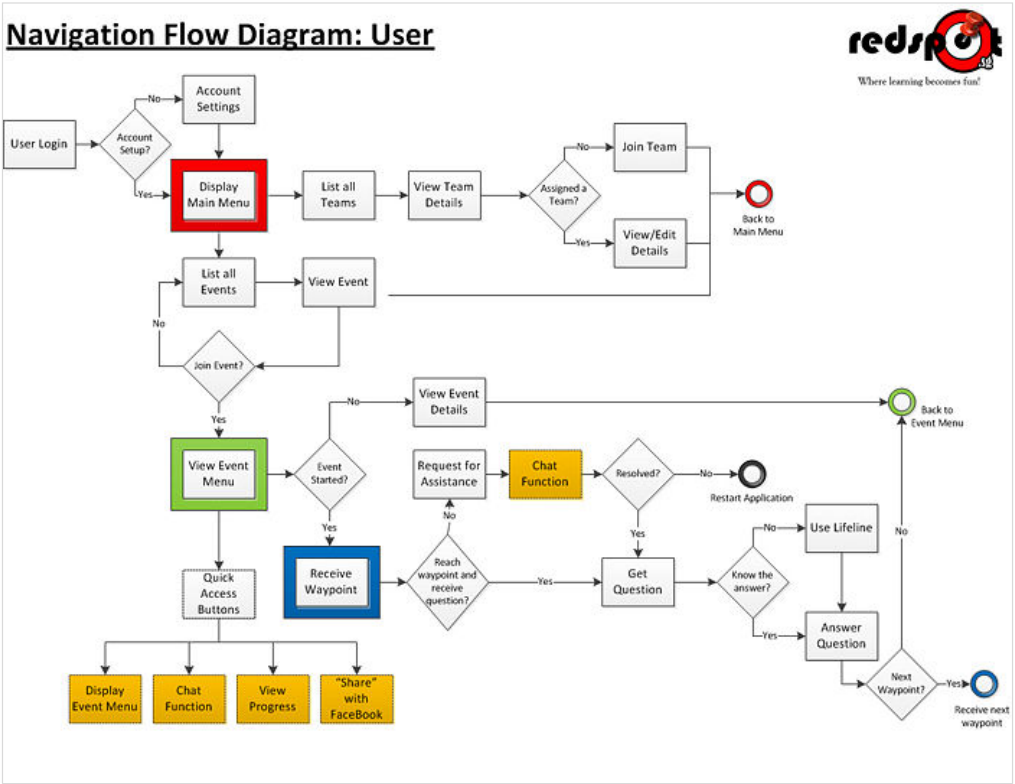
GUI proto and navigation





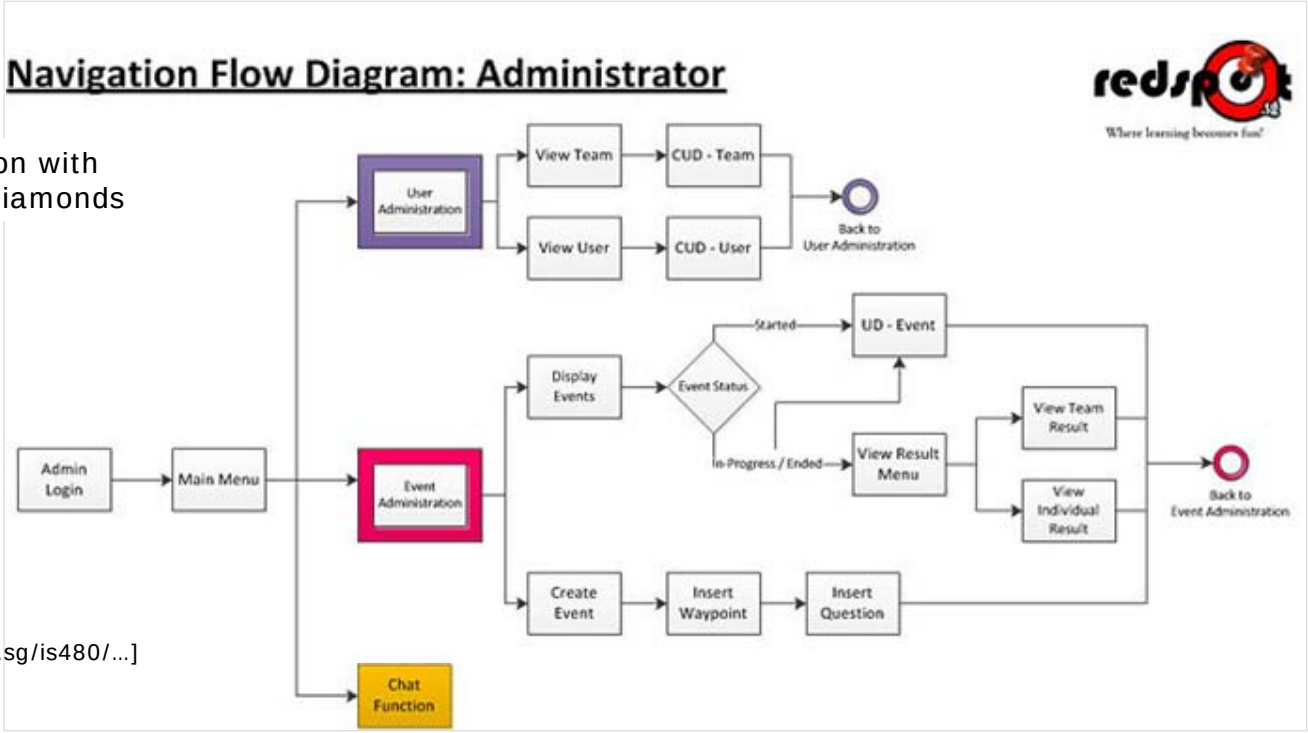
[s2713275.wordpress.com/2012/09/28/part-3-allsofts-web-site-design-wire-frames/]

Navigation with
Yes/No diamonds



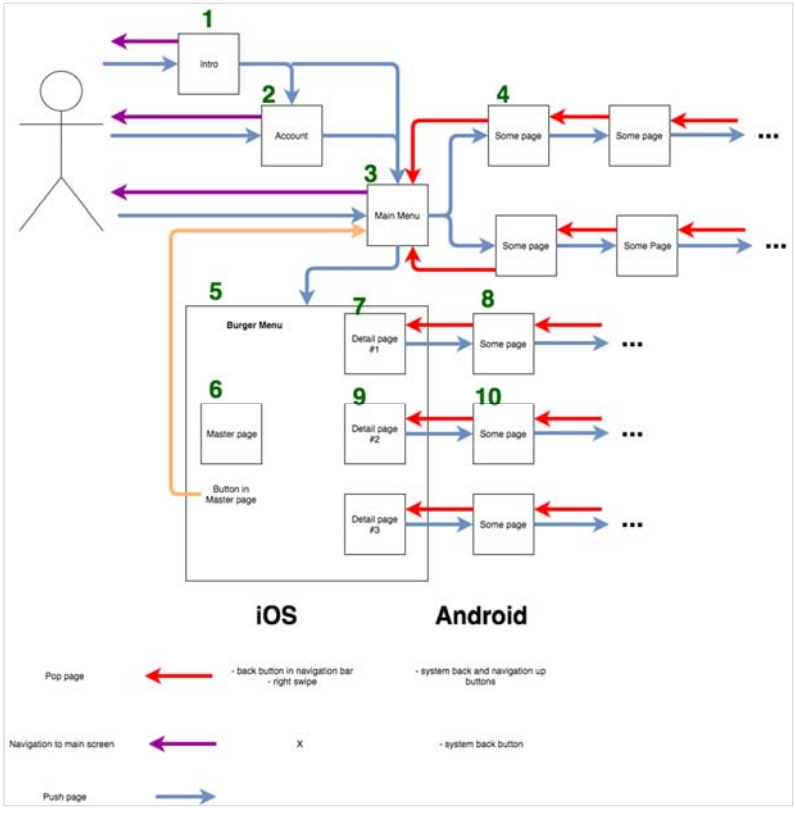
[wiki.smu.edu.sg/is480/...]

Navigation with
Yes/No diamonds



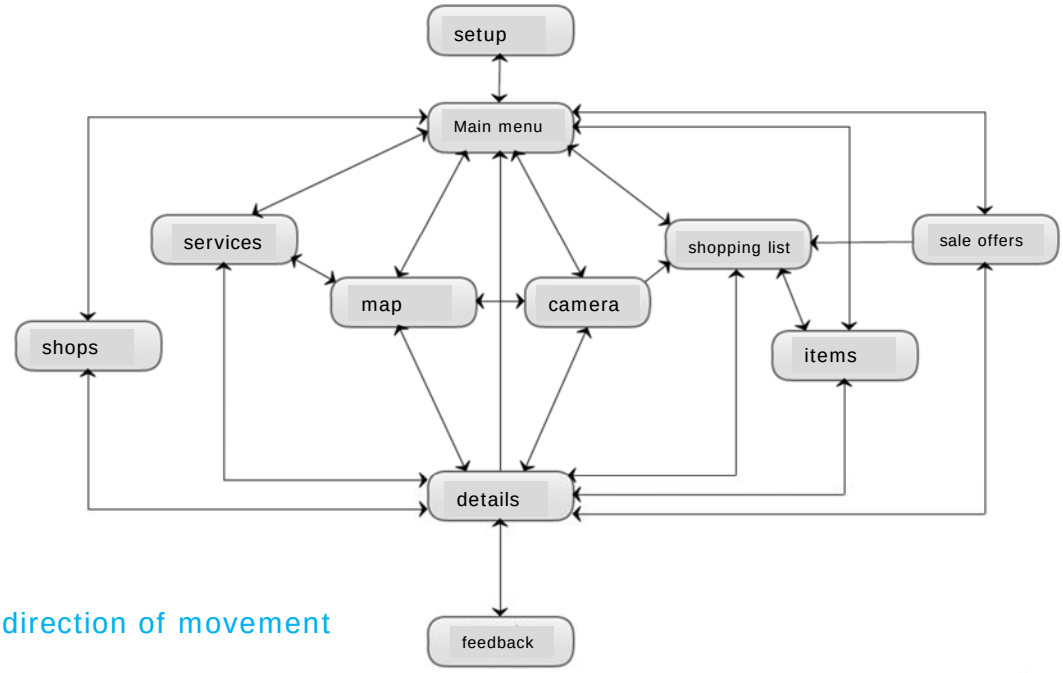
[wiki.smu.edu.sg/is480/…]

Navigation diagram



[i.stack.imgur.com/GF5nq.png]

Navigation diagram / chart / map



Arrows show direction of movement

Tampereen yliopisto
Tampere University

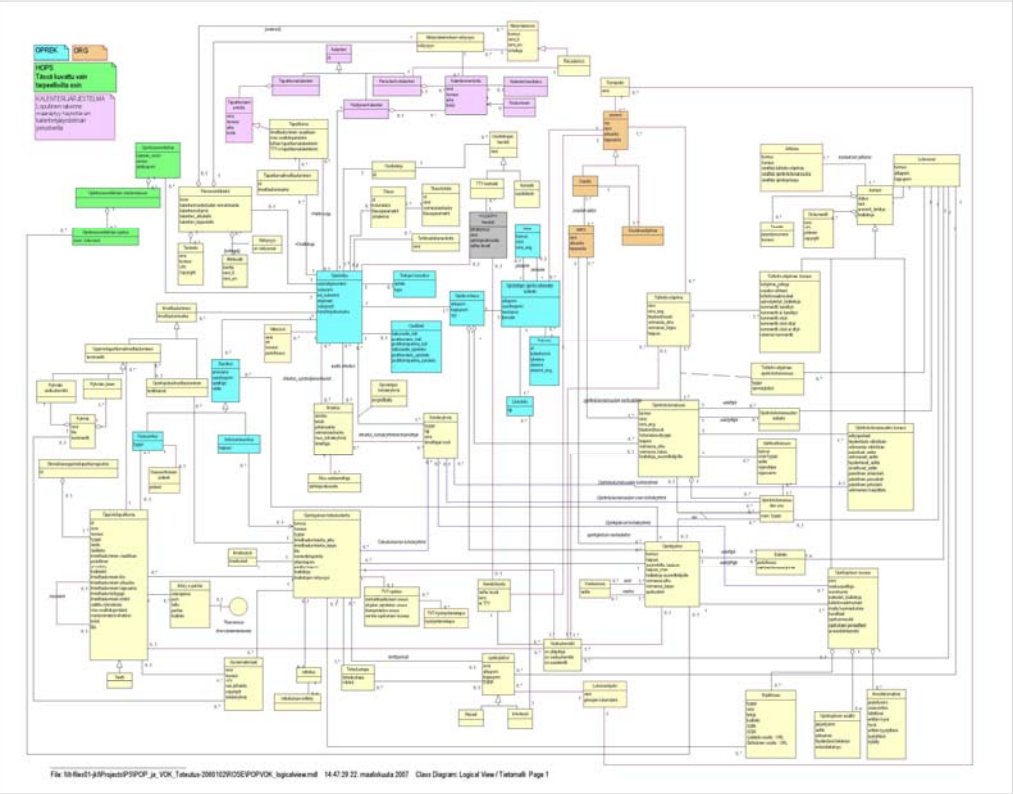
Many examples exist, e.g.

www.uml-diagrams.org/

See also **Additional material** at Moodle.

Tampereen yliopisto
Tampere University

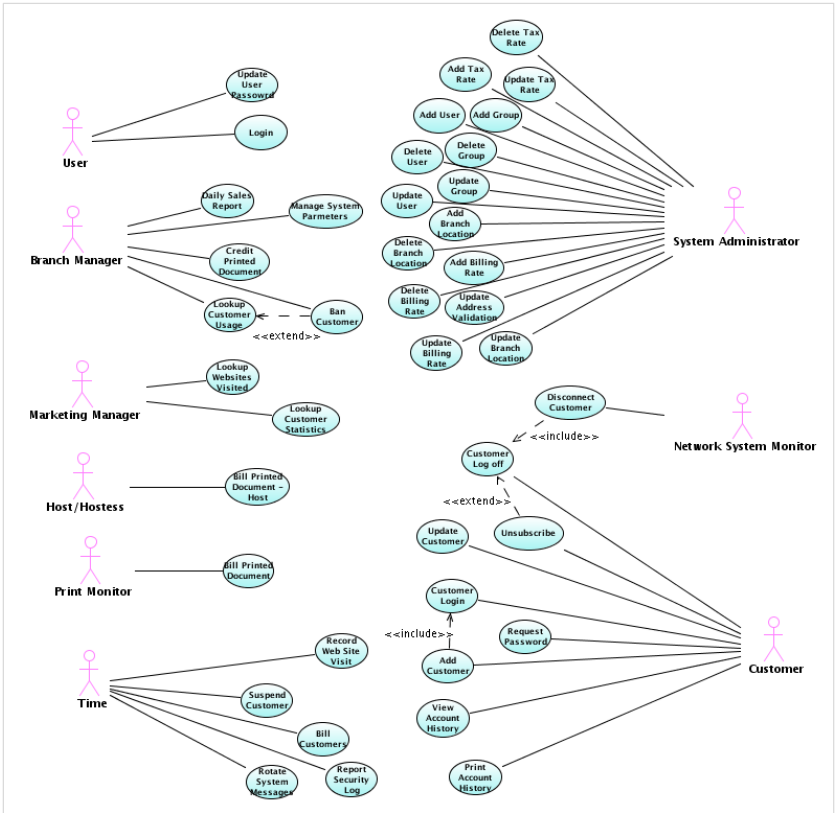
Too big diagram ?
Here:
POP/ROCK.
Split diagram or get larger display.



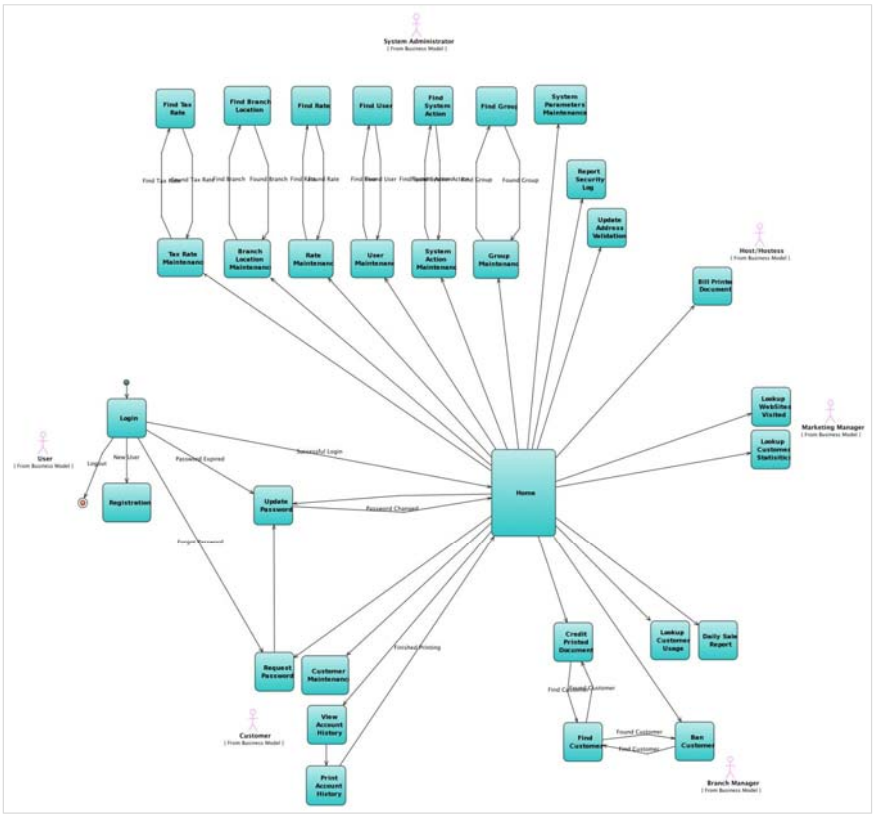
Tampereen yliopisto
Tampere University

Use Case diagram (International internet cafe)

[cafesource.sourceforge.net/phase1/]



Navigation diagram (International internet cafe)



[cafesource.sourceforge.net/phase1/]

Tampereen yliopisto
Tampere University

Highlights - What to remember

- **Diagrams** are needed to help at requirements gathering and specification phase. No just diagrams nor just **text** describes the system well; both are needed together supporting each other.
- **context** diagrams, **Use Case** diagrams and **navigation** charts are common diagrams at requirements phase
- there may be some variations in actual use (”UML-BUT”)
- remember also to make a **data dictionary** (= glossary)
- at your work, use whatever kind and style of diagrams that help